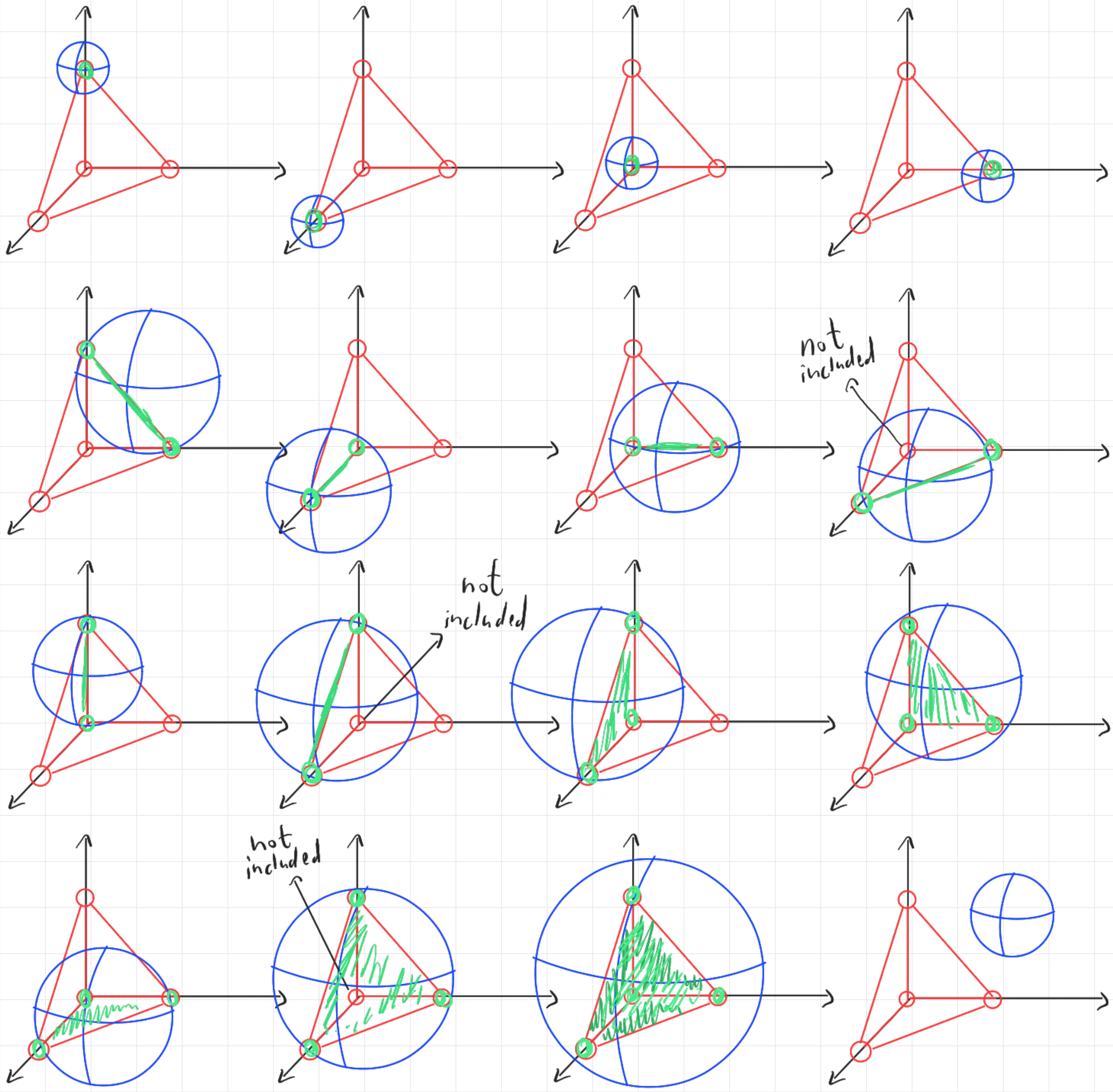
Problem 1.

a) What is the VC-dimension of the infinite set of uni-directional balls on three dimensional points? Prove your result.

1a) Given the set B of uni-directional 3D balls (assuming inside-green, and outside-red) over the 3D space, we shall prove that this set's VC-dimension is 4.

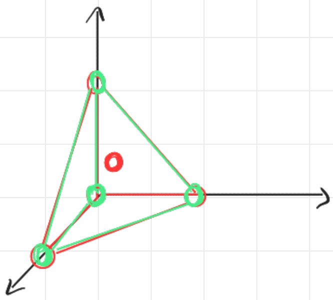
* Shattering an example of 4 points (and therefore proving that the set also shatters 1,2,3 points):



* Proof there's no 5 points B shatters:

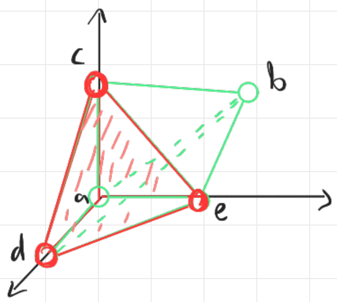
In this proof we will refer to the 2 general cases for 4 points and an additional point:

1) The additional point is inside an arbitrary convex hull made by the 4 points:



There is no way to exclude the red point since any of the infinite no. of points contained in the convex hull made by the 4 green points will always be contained in a Ball from B that contains them.

2) The opposite of one:



Due to the property of a ball's radius, which makes a ball grow at the same length at all directions from it's center, there is no way to include both a and b without also including c, d, e.
An ovoid, for example, which has the main properties of a ball but lacks the property above, can shatter these points.

b) Hypothesis class **C** contains both the infinite set of uni-directional balls and the infinite set of hyperplanes on three dimensional points. What is its VC-dimension? Prove your result.

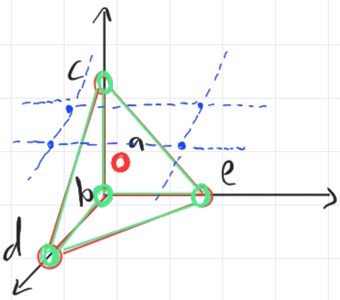
1b The VC dimension of C will also be 4.

To prove this, we will go over why uni-directional hyperplanes cannot shatter any 5 points.

Proving that the addition of the uni-directional hyperplanes does not help solve any of the 2 general cases shown in 1a, will suffice in order to conclude that they do not add classifying power to B's.

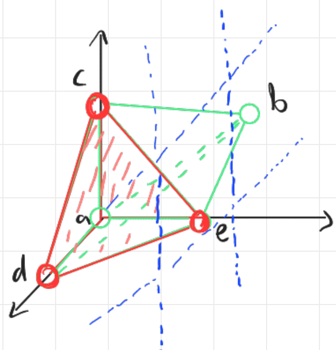
Since we've proved B cannot solve them, proving that they also can't will suffice.

1) The additional point is inside an arbitrary convex hull made by the 4 points:



A hyper plane in the 3D plane can cut at most 3 sides of a pyramid.
Therefore there is no way to place a hyperplane that has the red point at one side of it without also having at least one green point at the same side, therefore this case can't be solved for this classification of the points.

2) The opposite of one:



For the same reason as case 1, this case cannot be solved.

c) Hypothesis class **D** contains the infinite set of half-balls on three dimension points. Give an upper bound on its VC-dimension.

1c We can see that a half-ball is really a union between a ball and hyperplanes over the 3D space.

We've also seen throughout the lectures that the VC-Dimension of hyperplanes in the 3D space is 4 (because $dim = 2 \rightarrow D = 4$)

Since we've also showed in this exercise that B's VC-dimension is 4 we can utilize the Union of Hypothesis as seen in our course to create a union over these 2 sets of rules and get the following bound:

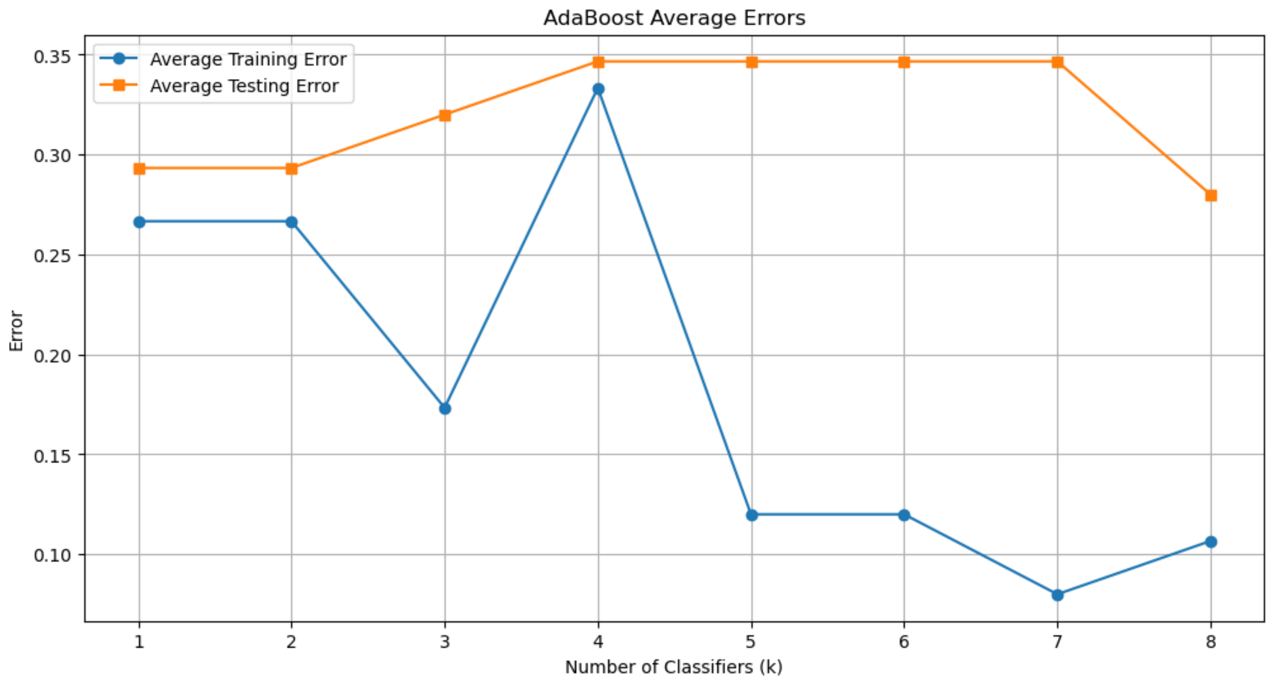
$$VC\text{-}dim(D) \leq 2 \cdot d \cdot s \cdot \log_2(3 \cdot s) = 2 \cdot 4 \cdot 2 \cdot \log_2(6) \sim 41.35$$

d) How many different labels can **D** give to 100 points? Give an upper bound.

1d Using Sauer's Lemma and the Union of Hypotheses, the bound Π of D over set P of 100 points is:

$$\Pi(D, P) \leq \left(\frac{e \cdot n}{d}\right)^{d \cdot s} = \left(\frac{100e}{4}\right)^{4 \cdot 2} = (25 \cdot e)^8$$

A+B) The following graph represents the training and testing errors achieved over 50 runs of executing AdaBoost, while each execution has 8 iterations over all line rules and points:



We have also gathered the average errors of iteration ‘i’ over all the executions of AdaBoost, and achieved as follows:

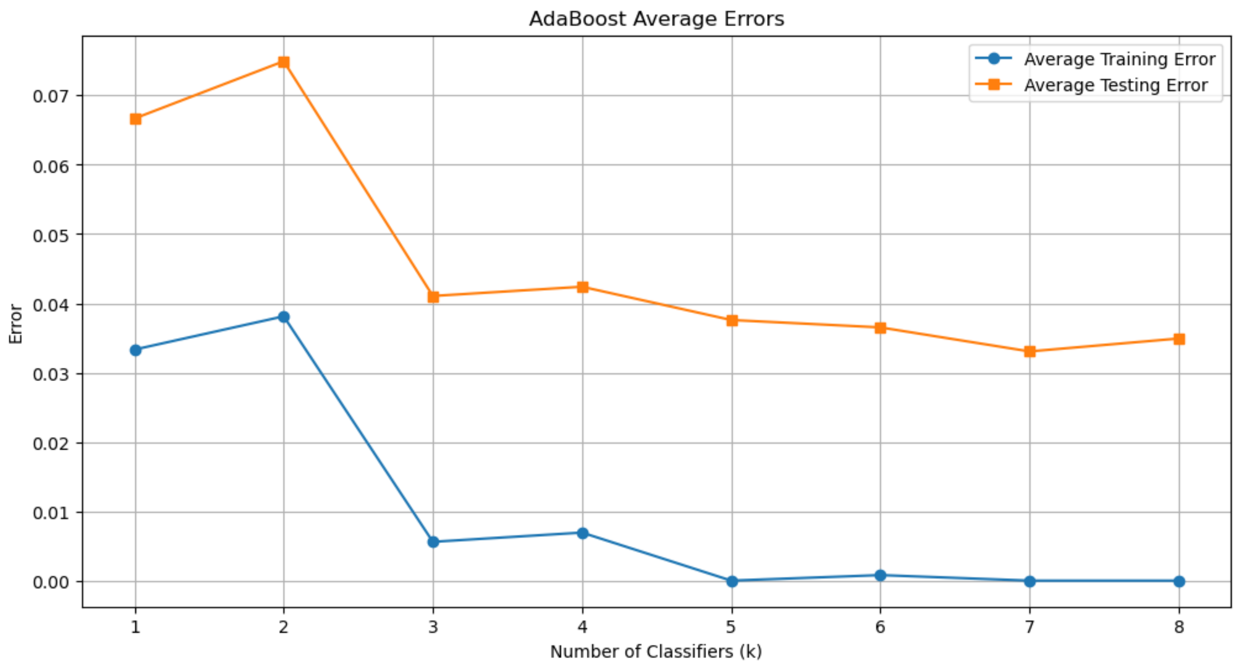
```
k = 1: Average Training error = 0.26, Average Testing error = 0.35
k = 2: Average Training error = 0.26, Average Testing error = 0.35
k = 3: Average Training error = 0.19, Average Testing error = 0.34
k = 4: Average Training error = 0.23, Average Testing error = 0.34
k = 5: Average Training error = 0.16, Average Testing error = 0.32
k = 6: Average Training error = 0.15, Average Testing error = 0.30
k = 7: Average Training error = 0.11, Average Testing error = 0.29
k = 8: Average Training error = 0.13, Average Testing error = 0.30
```

We can see that the training error gradually decreases as we advance in the number of iterations done, which is expected as the model fits the training data better over an increasing number of iterations

In the context of overfitting, we can see that the AdaBoost algorithm is effective. As we progress over the learning process of the model the true error decreases and is stabilized at around 0.3. This suggests that there is no clear evidence of overfitting in our model.

However, a true error of 0.3 is quite high, implying that the points do not distribute linearly and therefore the set of line rules does not make a good classifier in this case.

C) The following graph represents the same as before with the only change being the usage of circle classifiers instead:



And here also we gathered the average errors of iteration ‘i’ over all the executions of AdaBoost, and achieved as follows:

```
k = 1: Average Training error = 0.03, Average Testing error = 0.07
k = 2: Average Training error = 0.04, Average Testing error = 0.07
k = 3: Average Training error = 0.01, Average Testing error = 0.04
k = 4: Average Training error = 0.01, Average Testing error = 0.04
k = 5: Average Training error = 0.00, Average Testing error = 0.04
k = 6: Average Training error = 0.00, Average Testing error = 0.04
k = 7: Average Training error = 0.00, Average Testing error = 0.03
k = 8: Average Training error = 0.00, Average Testing error = 0.03
```

Same as before, we can see that the training error gradually decreases as we advance in the number of iterations done, implying that the model fits the training data better over an increasing number of iterations.

In the context of overfitting, we can see that the AdaBoost algorithm is effective. As we progress over the learning process of the model the true error decreases and is stabilized at around 0.04. This suggests that there is no clear evidence of overfitting in our model.

Since we have achieved a true error of about 0.04, this leads us to conclude that the points tend to be distributed in a circular fashion and therefore the set of circles make a strong classifier in this case.

D) Regarding the results of both of our testing phases, the usage of AdaBoost over the circular classifiers is evidenced to provide a stronger tool of classification over our dataset than with the set of lines.

This connects directly to the closing paragraph of C, which leads us to conclude that the points in our dataset are distributed circularly.